IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Appellants:	Gerard Chauvel, et al.	§	Confirmation No.:	1110
		§		
Serial No.:	10/632,216	§		
		§	Group Art Unit:	2183
Filed:	July 31, 2003	§		
		§		
For:	Micro-Sequence Execution In A Processor	§	Examiner:	Petranek, Andrew J
		§		

APPEAL BRIEF

Mail Stop Appeal Brief – Patents
Commissioner for Patents
PO Box 1450
Alexandria, VA 22313-1450

Date: November 16, 2007
Atty Docket No.: TI-35445 (1962-05415)

Sir:

Appellants hereby submit this Appeal Brief in connection with the above-identified application. A Notice of Appeal is filed concurrently herewith.

TABLE OF CONTENTS

I. REAL PARTY IN INTEREST

The real party in interest is Texas Instruments Inc., a Delaware corporation, having its principal place of business in Dallas, Texas. The Assignment from the inventors to Texas Instruments-France was recorded on July 31, 2003 at Reel/Frame 014356/0338, the Assignment from Texas Instruments-France to Texas Instruments Inc. was recorded on March 12, 2004 014421/0951.

II. RELATED APPEALS AND INTERFERENCES

Appellants are unaware of any related appeals or interferences.

III. STATUS OF THE CLAIMS

Originally filed claims:	1-23.
Claim cancellations:	None.
Added claims:	None.
Presently pending claims:	1-23.
Allowed claims:	None.
Presently appealed claims:	1-23.

IV. STATUS OF THE AMENDMENTS

No claims were amended after the final Office action dated August 17, 2007.

V. SUMMARY OF THE CLAIMED SUBJECT MATTER

The specification is directed to micro-sequence execution in a processor.[1] At least some of the various embodiments as in claim 1:

> A processor, comprising:
> fetch logic that retrieves instructions from memory[2];
> decode logic coupled to said fetch logic, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set[3]; and
> an active program counter selected as either a first program counter or a second program counter[4];
> wherein an instruction of the first instruction set is replaced by a micro-sequence comprising one or more instructions of the second instruction set and the active program counter switches between the first and second program counters based on a micro-sequence-active bit[5].

Other illustrative embodiments are as in claim 10:

> A method, comprising:
> fetching an instruction[6]; and
> determining whether said instruction is to be executed or replaced by a group of other instructions, the determining independent of the type of the instruction[7].

Yet still other illustrative embodiments are as in claim 15:

> A system, comprising:
> a first processor[8]; and
> a second processor coupled to said first processor[9], said second processor comprising:

---

[1] Specification Title.

[2] Specification page 8, paragraph [0019], lines 6-7. The balance of this Appeal Brief uses a shorthand notation for citations to the Specification in the form: ([page],[paragraph number], [lines]). Thus, this illustrative citation in the shorthand form reads (8, [0019], lines 6-7). *See also*, Figures 2, element 154.

[3] (10, [0022], lines 2-7); Figure 2, element 152.

[4] (14, [0031], lines 2-7); Figure 4, element 186 and 188

[5] (14, [0031], lines 7-15); Figure 4, element 152.

[6] (8, [0019], lines 6-7); Figure 2, element 154.

[7] (13, [0030], lines 1-8); Figure 2, element 152.

[8] (5, [0014], lines 2-4); Figure 1, element 102.

[9] (5, [0014], lines 2-4); Figure 1, element 104.

fetch logic that retrieves instructions from memory[10];
decode logic coupled to said fetch logic, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set[11]; and
an active program counter selected as either a first program counter or a second program counter[12];
wherein an instruction of the first instruction set is replaced by a micro-sequence comprising one or more instructions from the second instruction set and the active program counter switches between the first and second program counters based on a micro-sequence active bit[13].

Other illustrative embodiments are as in claim 18:

An electronic device, comprising:
decode logic that decodes instructions, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set[14]; and
a vector table comprising a plurality of entries, each entry corresponding to a separate instruction and including a first field indicating whether the corresponding instruction is to be executed by the electronic device or whether the instruction is to be replaced by a predetermined group of instructions stored in memory[15].

Yet still other illustrative embodiments are claim 22:

A processor, comprising:
decode logic that decodes instructions[16]; and
a means for[17] determining whether an instruction is to be executed or replaced by a micro-sequence of other instructions[18].

---

[10] (8, [0019], lines 6-7); Figure 2, element 154.

[11] (10, [0022], lines 2-7); Figure 2, element 152.

[12] (14, [0031], lines 2-7); Figure 4, element 186 and 188

[13] (14, [0031], lines 7-15); Figure 4, element 152.

[14] (10, [0022], lines 2-7); Figure 2, element 152.

[15] (11, [0026], lines 1-7 to 12, [0026], lines 1-4); Figure 4, element 162.

[16] (10, [0022], lines 2-7); Figure 2, element 152.

[17] This limitation is specifically identified as a means-plus-function limitation under 35 USC § 112, 6th paragraph.

[18] (13, [0030], lines 1-8); Figure 2, element 152.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Whether claims 10-14 are unpatentable under 35 USC § 112, 1st paragraph.

Whether claims 1-4, 7-9, and 19-20 are obvious under 35 USC § 103 over Seal et al. (U.S. Pat. No. 6,965,984, hereinafter Seal) in view of Gee et al. (U.S. Pat. No. 6,317,872, hereinafter Gee).

Whether claims 10 and 13 are anticipated under 35 USC § 102(e) by Gee.

Whether claims 10-12, 14 and 22-23 are obvious under 35 USC § 103 over Zaidi (U.S. Pat. No. 6,581,154, hereinafter Zaidi).

Whether claims 15-17 are obvious under 35 USC § 103 over Seal in view of Gee further in view of Greenberger (U.S. Pat. No. 6,092,179, hereinafter Greenberger).

Whether claims 18 and 21 are obvious under 35 USC § 103 over Seal.

VII. ARGUMENT

A. Section §112, first paragraph, Rejections

1. Claims 10-14

Claims 10-14 stand rejected as allegedly lacking enablement. In particular, the Office action states that one of ordinary skill in the art is not enabled by the specification to "determin[e] independent of the type of the instruction." Appellants respectfully traverse. With regard to the determination, Appellants direct the Board to the corresponding paragraph of the specification.

> In operation, the decode logic 152 uses a Bytecode from instructions 170 as an index into micro-sequence vector table 162. Once the decode logic 152 locates the indexed entry 164, the decode logic 152 examines the associated bit 168 to determine whether the Bytecode is to be replaced by a micro-sequence. If the bit 168 indicates that the Bytecode can be directly processed and executed by the JSM, then the instruction is so executed. If, however, the bit 168 indicates that the Bytecode is to be replaced by a micro-sequence, then the decode logic 152 preferably changes this instruction into a "NOP" and sets the micro-sequence-active bit (described above) in the status register R15… .[19]

In particular, the specification discloses that the decode logic fetches a Bytecode instruction from an instruction storage[20], then the decode logic uses the Bytecode as an index into the vector table to examine the associated bit.[21] Thus, the determination whether the Bytecode is to be replaced is made independent of the **type** of Bytecode fetched from the memory. In fact, the determination of the type of Bytecode fetched is only made after reading the entry in the vector table.[22]

In the Response to Arguments section of the Office Action dated August 17, 2007, the Office action states that "the type of the instruction being fetched and decoded has a direct effect on the control signals generated by the decoder… ."[23] Appellants respectfully traverse. Insofar as an instruction is executed, the Office Action is correct; however, as

---

[19] (13, [0030], lines 1-8); Figure 4, element 152 and 162.

[20] (11, [0025], lines 5-7); Figure 4, element 152.

[21] (13, [0030], lines 1-4); Figure 4, element 162.

[22] (13, [0030], lines 1-8); Figure 4, element 152 and 162.

[23] Office Action of August 17, 2007, page 19, last paragraph.

discussed in the immediately preceding paragraph, when the index indicates an instruction is to be replaced, the decoder does not get to the point where "the type of instruction being fetched and decoded has an direct effect on the control signals generated by the decoder… .", the decode logic changes the instruction into a NOP.[24] The result of the initial determination by the decode logic is whether the Bytecode should be executed by the JSM or replaced.[25] For this reason alone the Section 112 rejections should be withdrawn.

The Office Action goes further to state that "the element that truly determines whether an instruction is to be executed or replaced by micro-sequence is element 168 in figure 4."[26] Appellants respectfully traverse. The element 168 is a bit associated with each entry in the micro-sequence vector table.[27] A single bit, such as element 168, in a table or the table by itself is merely data and imparts no functionality. Thus, the interpretation that the element 168 determines whether to execute or replace is clearly improper, and is inconsistent with the specification which discloses that the **decode logic** uses the associated bit (element 168) in the vector table to determine whether to execute or replace the instruction.[28]

Based on the foregoing, Applicants respectfully submit that 35 USC 112, 1st paragraph lack of enablement rejection should be withdrawn.

Claims 10-14 also stand rejected for allegedly failing to comply with written description requirement. In particular, the Office action states that there is no support in the application at the time filing for the limitation "the determining independent of the type of the instruction." [29] Appellants respectfully traverse. The specification provides written description support, and the Appellants direct the Board to the corresponding paragraph of the specification.

---

[24] (13, [0030], lines 1-8); Figure 4, element 152 and 162.

[25] (13, [0030], lines 1-8); Figure 4, element 152 and 162

[26] Office Action of August 17, 2007, page 20, first paragraph.

[27] (11, [0026], lines 1-6); Figure 4, element 168.

[28] (13, [0030], lines 1-8); Figure 4, element 152 and 162.

[29] Office Action of August 17, 2007, page 4, first paragraph.

> In operation, the decode logic 152 uses a Bytecode from instructions 170 as an index into micro-sequence vector table 162. Once the decode logic 152 locates the indexed entry 164, the decode logic 152 examines the associated bit 168 to determine whether the Bytecode is to be replaced by a micro-sequence. If the bit 168 indicates that the Bytecode can be directly processed and executed by the JSM, then the instruction is so executed. If, however, the bit 168 indicates that the Bytecode is to be replaced by a micro-sequence, then the decode logic 152 preferably changes this instruction into a "NOP" and sets the micro-sequence-active bit (described above) in the status register R15… .[30]

Thus, the specification discloses that the decode logic fetches Bytecode instructions from an instruction storage,[31] then the decode logic uses a Bytecode as an index into the vector table to examine the associated bit.[32] Thus, the determination whether the Bytecode is to be replaced is made independent of the type of Bytecode fetched from the memory. In fact, the determination of the type of Bytecode fetched is only made after reading an entry in the vector table that indicates Bytecode should be directly executed.

Based on the foregoing, Applicants respectfully submit that 35 USC 112, 1st paragraph lack of written description requirement rejection be withdrawn.

B. Section § 103 Rejections over Seal and Gee.

1. Claims 1, 3, 4, 7-9, and 19-20

Claims 1, 3, 4, 7-9, and 19-20 stand rejected as allegedly obvious over Seal and Gee. Claim 1 is representative of this grouping of claims. The grouping should not be construed to mean the patentability of any of the claims may be determined in later actions (*e.g.*, actions before a court) based on the groupings. Rather, the presumption of 35 U.S.C. § 282 shall apply to each of these claims individually.

Seal's Figure 1 is reproduced immediately below for convenience of the discussion.

[30] (13, [0030], lines 1-8); Figure 4, element 152 and 162.

[31] (11, [0025], lines 5-7); Figure 4, element 152.

[32] (13, [0030], lines 1-4); Figure 4, element 162.

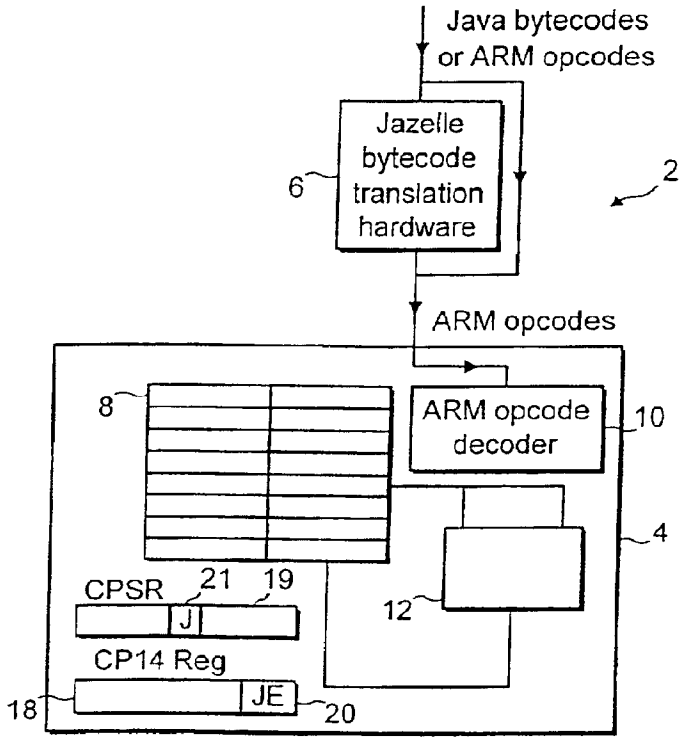

FIG. 1

Seal teaches an ARM processor 4, a bytecode translation hardware 6, and an instruction decoder 10.[33] Seal teaches that a stream of Java bytecodes are provided to bytecode translation hardware 6,[34] and the bytecode translation hardware 6 generates a series of ARM opcodes which are passed to the ARM opcode decoder 10.[35] Further, Seal teaches that ARM opcodes may be directly supplied to the ARM opcode decoder 10 by bypassing the bytecode translation hardware 6.[36] Thus, Seal appears to teach a decoder 10 that can decode only ARM instructions a Java bytecode translation hardware 6 that can only translate bytecodes.

Representative claim 1, by contrast, specifically recites "decode logic coupled to said fetch logic, the decode logic decodes instructions from a first instruction set and a

[33] Seal Col. 5, lines 60-66.

[34] Seal Col. 6, lines 10-14; Figure 1.

[35] Seal Col. 6, lines 10-29; Figure 1.

[36] Seal Col. 6, lines 10-29; Figure 1.

second instruction set, the second instruction set different than the first instruction set." Appellants respectfully submit Seal and Gee do not teach or fairly suggest such a processor. Seal appears to teach a decoder 10 that can decode only ARM instructions. Moreover, Seal's bytecode translation hardware 6 operates only on Java bytecodes. Thus, even if hypothetically the teachings of Gee are precisely as the Office action suggests (which Appellants do not admit), the references still fail to teach or fairly suggest a "decode logic coupled to said fetch logic, the decode logic decodes instructions from **a first instruction set and a second instruction set**, the second instruction set different than the first instruction set."

In the Response to Arguments section, the Office Action states that "it would be obvious to one of ordinary skill in the art…that there would have to be some sort of predecoding done to initially detect java bytecodes to send to element 6 or detect ARM opcodes that will be bypassed around element 6."[37] Appellants respectfully traverse. Seal teaches that register 19 includes a flag 21 that indicates active or inactive bytecode translation hardware.[38] The flag 21 in register 19 is set by a separate software control[39] and based on the flag 21, the system is provided either Java bytecodes or ARM instructions, but not a mixture. Thus, a "predecoding" is not necessary. Stated otherwise, it is not necessary to "initially detect java bytecodes to send to element 6 or detect ARM opcodes" as the system will only receive either Java bytecodes or ARM instructions.

Based on the foregoing, Appellants respectfully submit that the rejections of the claims in this grouping be reversed, and the claims set for issue.

2. Claim 2

Claim 2 stands rejected as allegedly obvious over Seal and Gee.

Seal teaches an ARM processor 4, a bytecode translation hardware 6, and an instruction decoder 10.[40] In particular, Seal teaches that a stream of Java bytecodes are provided to bytecode translation hardware 6,[41] and the bytecode translation hardware 6

[37] Office Action of August 17, 2007, page 20, last paragraph.

[38] Seal Col. 6, lines 2-4.

[39] Seal Col. 2, lines 53-60.

[40] Seal Col. 5, lines 60-66.

[41] Seal Col. 6, lines 10-14; Figure 1.

generates a series of ARM opcodes which are passed to the ARM opcode decoder 10.[42] In other cases, the bytecode translation hardware 6 is bypassed and ARM opcodes are directly provided to the ARM opcode decoder 10.[43] Seal further teaches that if a Java bytecode cannot be translated by the Java translation hardware 6 then the Java bytecode is used as an index to read a pointer in a table of pointers.[44] The pointer points to ARM code fragments that are decoded by the ARM opcode decoder 10 in place of the Java bytecode.[45] Thus, Seal appears to teach a table of pointers pointing to ARM code fragments, but the table does not include information which indicates whether the Java bytecode is to be replaced by the ARM code fragments; rather, the table is reached **only after** the translation hardware 6 is unable to translate the bytecodes. Moreover, Seal teaches that the table of pointers is accessible by the Java translation hardware 6 and not by the ARM opcode decoder 10.

Representative claim 2, by contrast, specifically recites "a vector table accessible by said decode logic, said vector table including information which specifies whether an instruction is to be replaced by a micro-sequence." Appellants respectfully submit Seal and Gee do not teach or fairly suggest such a processor. Seal appears to teach a table of pointers accessible by the Java translation hardware 6; however, the table of pointers does not include information which indicates whether the Java bytecodes are to be replaced by the ARM code fragments. Thus, even if hypothetically the teachings of Gee are precisely as the Office action suggests (which Appellants do not admit), the references still fail to teach or fairly suggest a "a vector table accessible by said decode logic, said vector table including information which specifies whether an instruction is to be replaced by a micro-sequence."

Based on the foregoing, Appellants respectfully submit that the rejections of the claim be reversed, and the claim set for issue.

---

[42] Seal Col. 6, lines 10-29; Figure 1.

[43] Seal Col. 6, lines 10-29; Figure 1.

[44] Seal Col. 6, lines 60-67.

[45] Seal Col. 6, lines 60-67.

3. Claims 5-6

Claims 5 and 6 stand rejected as allegedly obvious over Seal, Gee and Zaidi. Claims 5 and 6 are allowable for at least the same reasons as delineated in Section VII(B)(1).

C. Section § 102(e) Rejections over Gee

1. Claims 10 and 13

Claims 10 and 13 stand rejected as allegedly anticipated by Gee. Claim 10 is representative of this grouping of claims. The grouping should not be construed to mean the patentability of any of the claims may be determined in later actions (*e.g.*, actions before a court) based on the groupings. Rather, the presumption of 35 U.S.C. § 282 shall apply to each of these claims individually.

Gee is directed to a real time processor optimized for executing Java programs.[46] In particular, Gee teaches using an Advanced Architecture Microprocessor (AAMP) programmed to **simulat**e direct execution of Java bytecodes by using each bytecode as a pointer to a microinstruction sequence that performs the function of the bytecode.[47]

> The internal construction of the JEM processor is illustrated in FIG. 2. There are two levels of stored programs in the JEM system: one stored program is at the "micro" level using microinstructions in a control-store ROM 200**,** and the other stored program is at the "macro" level using bytecodes stored in a ROM code memory 104**.** The prefix "micro" will be used before elements which are involved with the microcoded portion of the processor to distinguish the processing of the "macro" bytecodes. In essence, **each bytecode is interpreted as a pointer to a sequence of microinstructions** which will actually be executed in place of the bytecode.[48]

Gee teaches two levels of stored programs, where **each** "macro" level bytecode is a pointer to a sequence of microinstructions. These microinstructions cause one or more elemental operations to occur in the machine.[49] Thus, Gee teaches that each and every

---

[46] Gee Title.

[47] Gee Col. 8, lines 57-59.

[48] Gee Col. 8, lines 49-59 (emphasis added).

[49] Gee Col. 8, lines 60-62.

bytecode points to a sequence of microinstructions that are executed in place of the bytecode.

Representative claim 10, by contrast, specifically recites, "determining whether said instruction is to be executed or replaced by a group of other instructions, the determining independent of the type of the instruction". Appellants respectfully submit that Gee fails to expressly or inherently teach such a method. Gee teaches that each and every bytecode points to a sequence of microinstructions that are executed in place of the bytecode. Thus, Gee fails to inherently or expressly teach "**determining whether said instruction is to be executed or replaced** by a group of other instructions, the determining independent of the type of the instruction."

In the Response to Arguments section, the Office Action states that "it's *[sic]* inherent that there is a determining step since each bytecode will be chosen to be replaced by a group of other instructions."[50] Appellants respectfully traverse. Appellants respectfully submit that since "each bytecode is interpreted as a pointer to a sequence of microinstructions which will actually be executed in place of the bytecode,"[51] determining is unnecessary. Stated otherwise, in accordance with Gee it is not required to decide whether to replace the fetched bytecode as the bytecode is **always** replaced by a group of other instructions.

Based on the foregoing, Appellants respectfully submit that the rejections of the claims of this grouping be reversed, and the claims set for issue.

2. Claims 11, 12 and 14

Claims 11, 12 and 14 stand rejected as allegedly obvious over Zaidi. Claims 11, 12 and 14 are allowable for at least the same reasons as delineated in Section VII(C)(1).

D. Section § 103 Rejections over Zaidi

1. Claims 10-12, 14 and 22-23

Claims 10-12, 14 and 22-23 stand rejected as allegedly obvious over Zaidi. Claim 10 is representative of this grouping of claims. The grouping should not be construed to mean the patentability of any of the claims may be determined in later

[50] Office Action August 17, 2007, page 21, second full paragraph.

[51] Gee Col. 8, lines 49-59 (emphasis added).

actions (*e.g.*, actions before a court) based on the groupings. Rather, the presumption of 35 U.S.C. § 282 shall apply to each of these claims individually.

Zaidi is directed to dynamically expanding and executing microcode routines utilizing a multi-level decoder.[52] Specifically, Zaidi teaches microcode as a collection of micro-operations.

> In this sense, microcode, in its basic form, is simply a collection of micro-opreations (uops), where uops are individual instructions or syllables.[53]

Zaidi also teaches a Super-uop (Suop).

> A Super-uop (Suop), on the other hand, is a uop than can be expanded or transformed into one or more simple, or regular uops by expansion logic. According to an embodiment, one Suop can represent multiple packed and a scalar instructions.[54]

Further, Zaidi teaches that a macro-instruction is transformed into uops or Suops and the processor executes the instructions contained in uop or Suop.

> Macro-instructions to be operated on by the processor are received by a micro-instruction sequencer (MIS) 200. The MIS logic transforms the macro-instructions into one or more uops and/or Suops… .[55]

If the macro-instruction is transformed into Suops, each Suop is expanded into one or more uops.

> The expander 210 uses associated indicator to expand each Suop into one or more uops to perform the scalar or packed operation indicated by the indicator.[56]

Thus, Zaidi appears to teach that Suops are merely a way of representing multiple uops. It follows that a Suop is not an instruction that can be executed; rather, a Suop merely a is representation of multiple instructions (uops).

---

[52] Zaidi Col. 2, lines 36-38.

[53] Zaidi Col. 2, lines 61-63.

[54] Zaidi Col. 2, lines 63-65.

[55] Zaidi Col. 3, lines 57-61.

[56] Zaidi Col. 4, lines 5-7.

Representative claim 10, by contrast, specifically recites, "fetching an instruction; and determining whether said instruction is to be executed or replaced by a group of other instructions." Appellants respectfully submit that Zaidi fails to teach or fairly suggest such a method. Zaidi teaches Suops as merely a way of representing multiple Uops and are not themselves executable. Thus, Zaidi fails to teach or fairly suggest that a "fetching an instruction; and determining whether said instruction is to be **executed or replaced by a group of other instructions**."

Based on the foregoing, Appellants respectfully submit that the rejections of the claims of this grouping be overturned, and the claims set for issue.

2. Claim 13

Claim 13 stands rejected as allegedly anticipated by Gee. Claim 13 is allowable for at least the same reasons as delineated in Section VII(D)(1).

E. Section § 103 Rejections over Seal, Gee and Greenberger

1. Claims 15 and 17

Claims 15 and 17 stand rejected as allegedly obvious over Seal, Gee and Greenberger. Claim 15 is representative of this grouping of claims. The grouping should not be construed to mean the patentability of any of the claims may be determined in later actions (*e.g.*, actions before a court) based on the groupings. Rather, the presumption of 35 U.S.C. § 282 shall apply to each of these claims individually.

Seal teaches an ARM processor 4, a bytecode translation hardware 6, and an instruction decoder 10.[57] In particular, Seal teaches that a stream of Java bytecodes are provided to bytecode translation hardware 6,[58] and the bytecode translation hardware 6 generates a series of ARM opcodes which are passed to the ARM opcode decoder 10.[59] In other cases, the bytecode translation hardware 6 is bypassed and ARM instructions are directly provided to the ARM opcode decoder 10.[60] Seal appears to teach a decoder 10 that can decode only ARM instructions and a Java bytecode translation hardware 6 that can only translate bytecodes.

---

[57] Seal Col. 5, lines 60-66.

[58] Seal Col. 6, lines 10-14; Figure 1.

[59] Seal Col. 6, lines 10-29; Figure 1.

Representative claim 15, by contrast, specifically recites "decode logic coupled to said fetch logic, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set." Appellants respectfully submit Seal, Gee and Greenberger do not teach or fairly suggest such a processor. Seal appears to teach a decoder 10 that can decode only ARM instructions. Moreover, Seal's bytecode translation hardware operates only on Java bytecodes. Thus, even if hypothetically the teachings of Gee and Greenberger are precisely as the Office action suggests (which Appellants do not admit), the references still fail to teach or fairly suggest that a "decode logic coupled to said fetch logic, the decode logic decodes instructions from **a first instruction set and a second instruction set**, the second instruction set different than the first instruction set."

Based on the foregoing, Appellants respectfully submit that the rejections of the claims in this grouping be reversed, and the claims set for issue.

2. Claim 16

Claim 16 stands rejected as allegedly obvious over Seal and Gee.

Seal teaches an ARM processor 4, a bytecode translation hardware 6, and an instruction decoder 10.[61] In particular, Seal teaches that a stream of Java bytecodes are provided to bytecode translation hardware 6,[62] and the bytecode translation hardware 6 generates a series of ARM opcodes which are passed to the ARM opcode decoder 10.[63] In other cases, the bytecode translation hardware 6 is bypassed and ARM opcodes are directly provided to the ARM opcode decoder 10.[64] Seal further teaches that if a Java bytecode cannot be translated by the Java translation hardware 6 then the Java bytecode is used as an index to read a pointer in a table of pointers.[65] The pointer points to ARM code fragments that are decoded by the ARM opcode decoder 10 in place of the Java

---

[60] Seal Col. 6, lines 10-29; Figure 1.

[61] Seal Col. 5, lines 60-66.

[62] Seal Col. 6, lines 10-14; Figure 1.

[63] Seal Col. 6, lines 10-29; Figure 1.

[64] Seal Col. 6, lines 10-29; Figure 1.

[65] Seal Col. 6, lines 60-67.

bytecode.[66] Thus, Seal appears to teach a table of pointers pointing to ARM code fragments, but the table does not include information which indicates whether the Java bytecode is to be replaced by the ARM code fragments; rather, the table is reached **only after** the translation hardware 6 is unable to translate the bytecodes. Moreover, Seal teaches that the table of pointers is accessible by the Java translation hardware 6 and not by the ARM opcode decoder 10.

Representative claim 16, by contrast, specifically recites "wherein said second processor further includes a micro-sequence vector table comprising a plurality of entries, each entry corresponding to a separate instruction and including a field that indicates whether the corresponding instruction is to be executed by the second processor or whether the instruction is to be replaced by a micro-sequence." Appellants respectfully submit Seal, Gee and Greenberger do not teach or fairly suggest such a processor. Seal appears to teach a table of pointers accessible by the Java translation hardware 6; however, the table of pointers does not include information which indicates whether the Java bytecodes are to be replaced by the ARM code fragments. Thus, even if hypothetically the teachings of Gee and Greenberger are precisely as the Office action suggests (which Appellants do not admit), the references still fail to teach or fairly suggest a "wherein said second processor further includes a micro-sequence vector table comprising a plurality of entries, **each entry corresponding to a separate instruction and including a field that indicates whether the corresponding instruction is to be executed by the second processor or whether the instruction is to be replaced by a micro-sequence**."

Based on the foregoing, Appellants respectfully submit that the rejections of the claims in this grouping be reversed, and the claims set for issue.

F. Section § 103 Rejections over Seal

1. Claims 18 and 21

Claims 18 and 21 stand rejected as allegedly obvious over Seal. Claim 18 is representative of this grouping of claims. The grouping should not be construed to mean the patentability of any of the claims may be determined in later actions (*e.g.*, actions

[66] Seal Col. 6, lines 60-67.

before a court) based on the groupings. Rather, the presumption of 35 U.S.C. § 282 shall apply to each of these claims individually.

Seal teaches an ARM processor 4, a bytecode translation hardware 6, and an instruction decoder 10.[67] In particular, Seal teaches that a stream of Java bytecodes are provided to bytecode translation hardware 6,[68] and the bytecode translation hardware 6 generates a series of ARM opcodes which are passed to the ARM opcode decoder 10.[69] In other cases, the bytecode translation hardware 6 is bypassed and ARM instructions are directly provided to the ARM opcode decoder 10.[70] Seal appears to teach a decoder 10 that can decode only ARM instructions and a Java bytecode translation hardware 6 that can only translate bytecodes.

Respresentative claim 18, by contrast, specifically recites "decode logic that decodes instructions, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set." Appellants respectfully submit Seal does not teach or fairly suggest such a device. Seal appears to teach a decoder 10 that can decode only ARM instructions. Moreover, Seal's bytecode translation hardware operates only on Java bytecodes. Thus, Seal fails to teach or fairly suggest "decode logic that decodes instructions, the decode logic decodes instructions from **a first instruction set and a second instruction set, the second instruction set different than the first instruction set**."

Based on the foregoing, Appellants respectfully submit that the rejections of the claims of this grouping be overturned, and the claims set for issue.

2. Claims 19-20

Claims 19-20 stands rejected as allegedly obvious over Seal and Gee. Claims 19-20 are allowable for at least the same reasons as delineated in Section VII(F)(1).

---

[67] Seal Col. 5, lines 60-66.

[68] Seal Col. 6, lines 10-14; Figure 1.

[69] Seal Col. 6, lines 10-29; Figure 1.

[70] Seal Col. 6, lines 10-29; Figure 1.

G. Conclusion

For the reasons stated above, Appellants respectfully submit that the Examiner erred in rejecting all pending claims. It is believed that no extensions of time or fees are required, beyond those that may otherwise be provided for in documents accompanying this paper. However, in the event that additional extensions of time are necessary to allow consideration of this paper, such extensions are hereby petitioned under 37 C.F.R. § 1.136(a), and any fees required (including fees for net addition of claims) are hereby authorized to be charged to the Texas Instruments, Inc. Deposit Account No. 20-0668.

Respectfully submitted,

/Utpal D. Shah/

Utpal D. Shah
PTO Reg. No. 60,047
CONLEY ROSE, P.C.
(512) 610-3440 (Phone)
(512) 610-3456 (Fax)
AGENT FOR APPELLANTS

VIII. CLAIMS APPENDIX

1. (Previously Presented) A processor, comprising:

fetch logic that retrieves instructions from memory;

decode logic coupled to said fetch logic, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set; and

an active program counter selected as either a first program counter or a second program counter;

wherein an instruction of the first instruction set is replaced by a micro-sequence comprising one or more instructions of the second instruction set and the active program counter switches between the first and second program counters based on a micro-sequence-active bit.

2. (Original) The processor of claim 1 further including a vector table accessible by said decode logic, said vector table including information which specifies whether an instruction is to be replaced by a micro-sequence.

3. (Original) The processor of claim 2 wherein the information is provided to the vector table from a block of memory accessible to the processor by an indirect addressing mode used in a repeat loop comprising at least one instruction.

4. (Original) The processor of claim 2 wherein the vector table comprises a plurality of entries and any one entry can be modified independently of the other entries.

5. (Original) The processor of claim 1 further including a micro-sequence vector table comprising a plurality of entries, each entry corresponding to a separate instruction and associated with a bit indicating whether the corresponding instruction is to be executed by the processor or whether the instruction is to be replaced by a micro-sequence.

6. (Previously Presented) The processor of claim 5 wherein at least some of the entries include a reference to a memory location in which a micro-sequence is stored depending if the associated bit indicates that the instruction is to be replaced by a micro-sequence.

7. (Original) The processor of claim 1 wherein the active program counter again switches between the first and second program counters when the micro-sequence is completed.

8. (Original) The processor of claim 1 wherein the second program counter is used to fetch and decode instructions comprising a micro-sequence and switching between the first and second program counters comprises switching from the first program counter to the second program counter and loading the second program counter with a starting address of the micro-sequence.

9. (Original) The processor of claim 1 wherein a plurality of instructions are replaceable by a corresponding micro-sequence.

10. (Previously Presented) A method, comprising:
fetching an instruction; and
determining whether said instruction is to be executed or replaced by a group of other instructions, the determining independent of the type of the instruction.

11. (Original) The method of claim 10 further including replacing the instruction with said group of other instructions.

12. (Original) The method of claim 10 wherein determining whether the instruction is to be executed or replaced includes determining a value of a bit associated with the instruction.

13. (Original) The method of claim 10 further including switching an active program counter between two program counters when replacing the instruction with the group of instructions.

14. (Previously Presented) The method of claim 10 further including programming a table to specify which instructions are to be executed directly and which instructions are to be replaced by a group of instructions.

15. (Previously Presented) A system, comprising:

a first processor; and

a second processor coupled to said first processor, said second processor comprising:

fetch logic that retrieves instructions from memory;

decode logic coupled to said fetch logic, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set; and

an active program counter selected as either a first program counter or a second program counter;

wherein an instruction of the first instruction set is replaced by a micro-sequence comprising one or more instructions from the second instruction set and the active program counter switches between the first and second program counters based on a micro-sequence active bit.

16. (Original) The system of claim 15 wherein said second processor further includes a micro-sequence vector table comprising a plurality of entries, each entry corresponding to a separate instruction and including a field that indicates whether the corresponding instruction is to be executed by the second processor or whether the instruction is to be replaced by a micro-sequence.

17. (Original) The system of claim 16 wherein each entry also includes a reference to a memory location in which a micro-sequence is stored depending on a value stored in the field.

18. (Previously Presented) An electronic device, comprising:

decode logic that decodes instructions, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set; and

a vector table comprising a plurality of entries, each entry corresponding to a separate instruction and including a first field indicating whether the corresponding instruction is to be executed by the electronic device or whether the instruction is to be replaced by a predetermined group of instructions stored in memory.

19. (Original) The electronic device of claim 18 further including an active program counter selected as either a first program counter or a second program counter, wherein an instruction is replaced by the group of instructions and the active program counter concurrently switches from the first to the second program counter.

20. (Original) The electronic device of claim 18 wherein upon switching the active program counter, the first program counter is incremented.

21. (Original) The electronic device of claim 18 wherein the group of instructions terminates with a predetermined instruction.

22. (Original) A processor, comprising:

decode logic that decodes instructions; and

a means for determining whether an instruction is to be executed or replaced by a micro-sequence of other instructions.

23. (Original) The processor of claim 22 further including a means for replacing the instruction with the micro-sequence.

IX. EVIDENCE APPENDIX

None.

X. RELATED PROCEEDINGS APPENDIX

None.